



CCCCCCCC	TTTTTTTT	SSSSSSSS	EEEEEEEEE	NN	NN	SSSSSSSS	EEEEEEEEE	RRRRRRRR	TTTTTTTT
CCCCCCCC	TTTTTTTT	SSSSSSSS	EEEEEEEEE	NN	NN	SSSSSSSS	EEEEEEEEE	RRRRRRRR	TTTTTTTT
CC	TT	SS	EE	NN	NN	SS	EE	RR	TT
CC	TT	SS	EE	NNNN	NN	SS	EE	RR	TT
CC	TT	SS	EE	NNNN	NN	SS	EE	RR	TT
CC	TT	SSSSSS	EEEEEEEEE	NN	NN	SSSSSS	EEEEEEEEE	RRRRRRRR	TT
CC	TT	SSSSSS	EEEEEEEEE	NN	NN	SSSSSS	EEEEEEEEE	RRRRRRRR	TT
CC	TT	SS	EE	NN	NNNN	SS	EE	RR	TT
CC	TT	SS	EE	NN	NNNN	SS	EE	RR	TT
CC	TT	SS	EE	NN	NN	SS	EE	RR	TT
CC	TT	SS	EE	NN	NN	SS	EE	RR	TT
CC	TT	SS	EE	NN	NN	SS	EE	RR	TT
CCCCCCCC	TT	SSSSSSSS	EEEEEEEEE	NN	NN	SSSSSSSS	EEEEEEEEE	RR	TT
CCCCCCCC	TT	SSSSSSSS	EEEEEEEEE	NN	NN	SSSSSSSS	EEEEEEEEE	RR	TT

....  
....  
....  
....

LL		SSSSSSSS
LL		SSSSSSSS
LL		SS
LL		SS
LL		SS
LL		SSSSSS
LL		SSSSSS
LL		SS
LL		SS
LL		SS
LLLLLLLLLL		SSSSSSSS
LLLLLLLLLL		SSSSSSSS

(2) 57  
(2) 217

DECLARATIONS  
CTSENSECHAR - Map READ\_CHAR into CHAR using VMS data

0000 1 .TITLE CTSENSERT - RTPAD/CTERM SENSE CHARACTERISTICS  
0000 2 .IDENT 'V04-000'  
0000 3 .ENABLE SUPPRESSION  
0000 4 .DISABLE GLOBAL  
0000 5  
0000 6 \*\*\*\*\*  
0000 7 \*  
0000 8 \* COPYRIGHT (c) 1978, 1980, 1982, 1984 BY  
0000 9 \* DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.  
0000 10 \* ALL RIGHTS RESERVED.  
0000 11 \*  
0000 12 \* THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED  
0000 13 \* ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE  
0000 14 \* INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER  
0000 15 \* COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY  
0000 16 \* OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY  
0000 17 \* TRANSFERRED.  
0000 18 \*  
0000 19 \* THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE  
0000 20 \* AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT  
0000 21 \* CORPORATION.  
0000 22 \*  
0000 23 \* DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS  
0000 24 \* SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.  
0000 25 \*  
0000 26 \*  
0000 27 \*\*\*\*\*  
0000 28 :  
0000 29 :  
0000 30 :++  
0000 31 :  
0000 32 : FACILITY:  
0000 33 : CTERM Remote terminal protocol driver  
0000 34 :  
0000 35 : ABSTRACT:  
0000 36 :  
0000 37 : This module is called to map a characteristics item list received  
0000 38 : from the net into VMS terminal characteristics.  
0000 39 :  
0000 40 : ENVIRONMENT:  
0000 41 :  
0000 42 :  
0000 43 :  
0000 44 :  
0000 45 :--  
0000 46 :  
0000 47 : AUTHOR: Jake VanNoy, CREATION DATE: 23-Aug-1982  
0000 48 :  
0000 49 : MODIFIED BY:  
0000 50 :  
0000 51 : V03-001 JLV0297 Jake VanNoy 28-JUL-1983  
0000 52 : Change characteristics symbols.  
0000 53 :  
0000 54 :\*\*  
0000 55 :  
0000 56 :  
0000 57 :  
0000 58 :  
0000 59 :  
0000 60 :  
0000 61 :  
0000 62 :  
0000 63 :  
0000 64 :  
0000 65 :  
0000 66 :  
0000 67 :  
0000 68 :  
0000 69 :  
0000 70 :  
0000 71 :  
0000 72 :  
0000 73 :  
0000 74 :  
0000 75 :  
0000 76 :  
0000 77 :  
0000 78 :  
0000 79 :  
0000 80 :  
0000 81 :  
0000 82 :  
0000 83 :  
0000 84 :  
0000 85 :  
0000 86 :  
0000 87 :  
0000 88 :  
0000 89 :  
0000 90 :  
0000 91 :  
0000 92 :  
0000 93 :  
0000 94 :  
0000 95 :  
0000 96 :  
0000 97 :  
0000 98 :  
0000 99 :  
0000 100 :  
0000 101 :  
0000 102 :  
0000 103 :  
0000 104 :  
0000 105 :  
0000 106 :  
0000 107 :  
0000 108 :  
0000 109 :  
0000 110 :  
0000 111 :  
0000 112 :  
0000 113 :  
0000 114 :  
0000 115 :  
0000 116 :  
0000 117 :  
0000 118 :  
0000 119 :  
0000 120 :  
0000 121 :  
0000 122 :  
0000 123 :  
0000 124 :  
0000 125 :  
0000 126 :  
0000 127 :  
0000 128 :  
0000 129 :  
0000 130 :  
0000 131 :  
0000 132 :  
0000 133 :  
0000 134 :  
0000 135 :  
0000 136 :  
0000 137 :  
0000 138 :  
0000 139 :  
0000 140 :  
0000 141 :  
0000 142 :  
0000 143 :  
0000 144 :  
0000 145 :  
0000 146 :  
0000 147 :  
0000 148 :  
0000 149 :  
0000 150 :  
0000 151 :  
0000 152 :  
0000 153 :  
0000 154 :  
0000 155 :  
0000 156 :  
0000 157 :  
0000 158 :  
0000 159 :  
0000 160 :  
0000 161 :  
0000 162 :  
0000 163 :  
0000 164 :  
0000 165 :  
0000 166 :  
0000 167 :  
0000 168 :  
0000 169 :  
0000 170 :  
0000 171 :  
0000 172 :  
0000 173 :  
0000 174 :  
0000 175 :  
0000 176 :  
0000 177 :  
0000 178 :  
0000 179 :  
0000 180 :  
0000 181 :  
0000 182 :  
0000 183 :  
0000 184 :  
0000 185 :  
0000 186 :  
0000 187 :  
0000 188 :  
0000 189 :  
0000 190 :  
0000 191 :  
0000 192 :  
0000 193 :  
0000 194 :  
0000 195 :  
0000 196 :  
0000 197 :  
0000 198 :  
0000 199 :  
0000 200 :  
0000 201 :  
0000 202 :  
0000 203 :  
0000 204 :  
0000 205 :  
0000 206 :  
0000 207 :  
0000 208 :  
0000 209 :  
0000 210 :  
0000 211 :  
0000 212 :  
0000 213 :  
0000 214 :  
0000 215 :  
0000 216 :  
0000 217 :  
0000 218 :  
0000 219 :  
0000 220 :  
0000 221 :  
0000 222 :  
0000 223 :  
0000 224 :  
0000 225 :  
0000 226 :  
0000 227 :  
0000 228 :  
0000 229 :  
0000 230 :  
0000 231 :  
0000 232 :  
0000 233 :  
0000 234 :  
0000 235 :  
0000 236 :  
0000 237 :  
0000 238 :  
0000 239 :  
0000 240 :  
0000 241 :  
0000 242 :  
0000 243 :  
0000 244 :  
0000 245 :  
0000 246 :  
0000 247 :  
0000 248 :  
0000 249 :  
0000 250 :  
0000 251 :  
0000 252 :  
0000 253 :  
0000 254 :  
0000 255 :  
0000 256 :  
0000 257 :  
0000 258 :  
0000 259 :  
0000 260 :  
0000 261 :  
0000 262 :  
0000 263 :  
0000 264 :  
0000 265 :  
0000 266 :  
0000 267 :  
0000 268 :  
0000 269 :  
0000 270 :  
0000 271 :  
0000 272 :  
0000 273 :  
0000 274 :  
0000 275 :  
0000 276 :  
0000 277 :  
0000 278 :  
0000 279 :  
0000 280 :  
0000 281 :  
0000 282 :  
0000 283 :  
0000 284 :  
0000 285 :  
0000 286 :  
0000 287 :  
0000 288 :  
0000 289 :  
0000 290 :  
0000 291 :  
0000 292 :  
0000 293 :  
0000 294 :  
0000 295 :  
0000 296 :  
0000 297 :  
0000 298 :  
0000 299 :  
0000 300 :  
0000 301 :  
0000 302 :  
0000 303 :  
0000 304 :  
0000 305 :  
0000 306 :  
0000 307 :  
0000 308 :  
0000 309 :  
0000 310 :  
0000 311 :  
0000 312 :  
0000 313 :  
0000 314 :  
0000 315 :  
0000 316 :  
0000 317 :  
0000 318 :  
0000 319 :  
0000 320 :  
0000 321 :  
0000 322 :  
0000 323 :  
0000 324 :  
0000 325 :  
0000 326 :  
0000 327 :  
0000 328 :  
0000 329 :  
0000 330 :  
0000 331 :  
0000 332 :  
0000 333 :  
0000 334 :  
0000 335 :  
0000 336 :  
0000 337 :  
0000 338 :  
0000 339 :  
0000 340 :  
0000 341 :  
0000 342 :  
0000 343 :  
0000 344 :  
0000 345 :  
0000 346 :  
0000 347 :  
0000 348 :  
0000 349 :  
0000 350 :  
0000 351 :  
0000 352 :  
0000 353 :  
0000 354 :  
0000 355 :  
0000 356 :  
0000 357 :  
0000 358 :  
0000 359 :  
0000 360 :  
0000 361 :  
0000 362 :  
0000 363 :  
0000 364 :  
0000 365 :  
0000 366 :  
0000 367 :  
0000 368 :  
0000 369 :  
0000 370 :  
0000 371 :  
0000 372 :  
0000 373 :  
0000 374 :  
0000 375 :  
0000 376 :  
0000 377 :  
0000 378 :  
0000 379 :  
0000 380 :  
0000 381 :  
0000 382 :  
0000 383 :  
0000 384 :  
0000 385 :  
0000 386 :  
0000 387 :  
0000 388 :  
0000 389 :  
0000 390 :  
0000 391 :  
0000 392 :  
0000 393 :  
0000 394 :  
0000 395 :  
0000 396 :  
0000 397 :  
0000 398 :  
0000 399 :  
0000 400 :  
0000 401 :  
0000 402 :  
0000 403 :  
0000 404 :  
0000 405 :  
0000 406 :  
0000 407 :  
0000 408 :  
0000 409 :  
0000 410 :  
0000 411 :  
0000 412 :  
0000 413 :  
0000 414 :  
0000 415 :  
0000 416 :  
0000 417 :  
0000 418 :  
0000 419 :  
0000 420 :  
0000 421 :  
0000 422 :  
0000 423 :  
0000 424 :  
0000 425 :  
0000 426 :  
0000 427 :  
0000 428 :  
0000 429 :  
0000 430 :  
0000 431 :  
0000 432 :  
0000 433 :  
0000 434 :  
0000 435 :  
0000 436 :  
0000 437 :  
0000 438 :  
0000 439 :  
0000 440 :  
0000 441 :  
0000 442 :  
0000 443 :  
0000 444 :  
0000 445 :  
0000 446 :  
0000 447 :  
0000 448 :  
0000 449 :  
0000 450 :  
0000 451 :  
0000 452 :  
0000 453 :  
0000 454 :  
0000 455 :  
0000 456 :  
0000 457 :  
0000 458 :  
0000 459 :  
0000 460 :  
0000 461 :  
0000 462 :  
0000 463 :  
0000 464 :  
0000 465 :  
0000 466 :  
0000 467 :  
0000 468 :  
0000 469 :  
0000 470 :  
0000 471 :  
0000 472 :  
0000 473 :  
0000 474 :  
0000 475 :  
0000 476 :  
0000 477 :  
0000 478 :  
0000 479 :  
0000 480 :  
0000 481 :  
0000 482 :  
0000 483 :  
0000 484 :  
0000 485 :  
0000 486 :  
0000 487 :  
0000 488 :  
0000 489 :  
0000 490 :  
0000 491 :  
0000 492 :  
0000 493 :  
0000 494 :  
0000 495 :  
0000 496 :  
0000 497 :  
0000 498 :  
0000 499 :  
0000 500 :  
0000 501 :  
0000 502 :  
0000 503 :  
0000 504 :  
0000 505 :  
0000 506 :  
0000 507 :  
0000 508 :  
0000 509 :  
0000 510 :  
0000 511 :  
0000 512 :  
0000 513 :  
0000 514 :  
0000 515 :  
0000 516 :  
0000 517 :  
0000 518 :  
0000 519 :  
0000 520 :  
0000 521 :  
0000 522 :  
0000 523 :  
0000 524 :  
0000 525 :  
0000 526 :  
0000 527 :  
0000 528 :  
0000 529 :  
0000 530 :  
0000 531 :  
0000 532 :  
0000 533 :  
0000 534 :  
0000 535 :  
0000 536 :  
0000 537 :  
0000 538 :  
0000 539 :  
0000 540 :  
0000 541 :  
0000 542 :  
0000 543 :  
0000 544 :  
0000 545 :  
0000 546 :  
0000 547 :  
0000 548 :  
0000 549 :  
0000 550 :  
0000 551 :  
0000 552 :  
0000 553 :  
0000 554 :  
0000 555 :  
0000 556 :  
0000 557 :  
0000 558 :  
0000 559 :  
0000 560 :  
0000 561 :  
0000 562 :  
0000 563 :  
0000 564 :  
0000 565 :  
0000 566 :  
0000 567 :  
0000 568 :  
0000 569 :  
0000 570 :  
0000 571 :  
0000 572 :  
0000 573 :  
0000 574 :  
0000 575 :  
0000 576 :  
0000 577 :  
0000 578 :  
0000 579 :  
0000 580 :  
0000 581 :  
0000 582 :  
0000 583 :  
0000 584 :  
0000 585 :  
0000 586 :  
0000 587 :  
0000 588 :  
0000 589 :  
0000 590 :  
0000 591 :  
0000 592 :  
0000 593 :  
0000 594 :  
0000 595 :  
0000 596 :  
0000 597 :  
0000 598 :  
0000 599 :  
0000 600 :  
0000 601 :  
0000 602 :  
0000 603 :  
0000 604 :  
0000 605 :  
0000 606 :  
0000 607 :  
0000 608 :  
0000 609 :  
0000 610 :  
0000 611 :  
0000 612 :  
0000 6

```
0000 57 .SBTTL DECLARATIONS
0000 58 :
0000 59 : INCLUDE FILES
0000 60 :
0000 61 $DCDEF
0000 62 $SSDEF
0000 63 $TTDEF
0000 64 $TT2DEF
0000 65 $TSADEF
0000 66 :
0000 67 :
0000 68 : MACROS:
0000 69 :
0000 70 :
0000 71 .MACRO CHAR LABEL
0000 72 .WORD char_type!char_count
0000 73 .WORD LABEL-SENSE_ROUTINES
0000 74 char_count = char_count + 1
0000 75 .ENDM CHAR
0000 76 :
0000 77 .MACRO BAUDTAB BPS
0000 78 .WORD BPS
0000 79 .ENDM BAUDTAB
0000 80 :
0000 81 .MACRO TERMTAB TERM,VALUE
0000 82 .BYTE VALUE
0000 83 .ASCII /TERM/
0000 84 .ENDM TERMTAB
0000 85 :
0000 86 .MACRO SENSETAB CHAR
0000 87 .WORD char_type!char
0000 88 .ENDM SENSETAB
0000 89 :
0000 90 :
0000 91 :
0000 92 : EQUATED SYMBOLS:
0000 93 :
0000 94 :
00000000 0000 95 physical = ch$C_physical@8 : 0 leftshifted 1 byte
00000100 0000 96 logical = ch$C_logical@8 : 1 leftshifted 1 byte
00000200 0000 97 cterm = ch$C_cterm@8 : 2 leftshifted 1 byte
00000000 0000 98 :
00000000 0000 99 TT_BUF = 0
00000004 0000 100 TT_CHAR1 = 4
00000008 0000 101 TT_CHAR2 = 8
00000000 0000 102 TT_IOSB = 0
00000000 0000 103 :
00000000 0000 104 :
00000000 0000 105 : OWN STORAGE:
00000000 0000 106 :
00000000 0000 107 :
00000000 0000 108 .PSECT RTPAD,NOWRT
00000000 0000 109 :
00000000 0000 110 SENSE_TABLE:
00000000 0000 111 :
00000000 0000 112 char_type = physical
00000001 0000 113 char_count = 1
```

0000	114	CHAR	INPUT SPEED	:	1
0004	115	CHAR	OUTPUT SPEED	:	2
0008	116	CHAR	CHARACTER SIZE	:	3
000C	117	CHAR	PARITY_ENABLE	:	4
0010	118	CHAR	PARITY_TYPE	:	5
0014	119	CHAR	MODEM_PRESENT	:	6
0018	120	CHAR	AUTOBAUD_DETECT	:	7
001C	121	CHAR	MANAGEMENT_GUARANTEED	:	8
0020	122	CHAR	SWITCH_CHAR_1	:	9
0024	123	CHAR	SWITCH_CHAR_2	:	10
0028	124				
00000100	0028	125	char_type = logical		
00000001	0028	126	char_count = 1		
0028	127				
0028	128	CHAR	MODE WRITING_ALLOWED	:	1
002C	129	CHAR	TERMINAL_TYPE	:	2
0030	130	CHAR	TERMINAL_SUBTYPE	:	3
0034	131	CHAR	OUTPUT_FLOW_CONTROL	:	4
0038	132	CHAR	OUTPUT_PAGE_STOP	:	5
003C	133	CHAR	FLOW_CHAR_PASS_THRU	:	6
0040	134	CHAR	INPUT_FLOW_CONTROL	:	7
0044	135	CHAR	LOSS_NOTIFICATION	:	8
0048	136	CHAR	LINE_WIDTH	:	9
004C	137	CHAR	PAGE_LENGTH	:	10
0050	138	CHAR	STOP_LENGTH	:	11
0054	139	CHAR	CR_FILL	:	12
0058	140	CHAR	LF_FILL	:	13
005C	141	CHAR	WRAP	:	14
0060	142	CHAR	HORIZONTAL_TAB	:	15
0064	143	CHAR	VERTICAL_TAB	:	16
0068	144	CHAR	FORM_FEED	:	17
006C	145				
00000200	006C	146	char_type = cterm		
00000001	006C	147	char_count = 1		
006C	148				
006C	149	CHAR	IGNORE_INPUT	:	1
0070	150	CHAR	CHAR_ATTRIBUTES	:	2
0074	151	CHAR	CONTROL_O_PASS_THRU	:	3
0078	152	CHAR	RAISE_INPUT	:	4
007C	153	CHAR	NORMAL_ECHO	:	5
0080	154	CHAR	REQUEST_PROCESSING	:	6
0080	155	CHAR	INPUT_ESCAPE_ENABLE	:	7
0084	156	CHAR	OUTPUT_ESCAPE_ENABLE	:	8
0088	157	CHAR	INPUT_COUNT_STATE	:	9
008C	158	CHAR	AUTO_PROMPT	:	10
0090	159	CHAR	ERROR_PROCESSING	:	11
0094	160				
FFFF	0094	161	.WORD -1		: End of list (any negative number)
0096	162				
0096	163	SENSE_BAUD:			: Table must be in ascending order
0096	164	BAUDTAB 0		:	0
0098	165	BAUDTAB 50		:	1
009A	166	BAUDTAB 75		:	2
009C	167	BAUDTAB 110		:	3
009E	168	BAUDTAB 134		:	4
00A0	169	BAUDTAB 150		:	5
00A2	170	BAUDTAB 300		:	6

00A4	171	BAUDTAB 600	: 7
00A6	172	BAUDTAB 1200	: 8
00A8	173	BAUDTAB 1800	: 9
00AA	174	BAUDTAB 2000	: 10
00AC	175	BAUDTAB 2400	: 11
00AE	176	BAUDTAB 3600	: 12
00B0	177	BAUDTAB 4800	: 13
00B2	178	BAUDTAB 7200	: 14
00B4	179	BAUDTAB 9600	: 15
00B6	180	BAUDTAB 19200	: 16
00B8	181		
00B8	182	TERM_TABLE:	: Table is in order for most likely first
00B8	183	TERMTAB VT100, TTS_VT100	
00BF	184	TERMTAB VT101, TTS_VT101	
00C6	185	TERMTAB VT102, TTS_VT102	
00CD	186	TERMTAB VT105, TTS_VT105	
00D4	187	TERMTAB VT125, TTS_VT125	
00DB	188	TERMTAB VT131, TTS_VT131	
00E2	189	TERMTAB VT132, TTS_VT132	
00E9	190	TERMTAB VT52, TTS_VT52	
00EF	191	TERMTAB VT05, TTS_VT05	
00F5	192	TERMTAB VK100, TTS_VK100	
00FC	193	TERMTAB VT173, TTS_VT173	
0103	194	TERMTAB TTS_FT1	
0103	195	TERMTAB TTS_FT2	
0103	196	TERMTAB TTS_FT3	
0103	197	TERMTAB TTS_FT4	
0103	198	TERMTAB TTS_FT5	
0103	199	TERMTAB TTS_FT6	
0103	200	TERMTAB TTS_FT7	
0103	201	TERMTAB TTS_FT8	
0103	202	TERMTAB LA36, TTS_LA36	
0109	203	TERMTAB LA12, TTS_LA12	
010F	204	TERMTAB LA34, TTS_LA34	
0115	205	TERMTAB LA38, TTS_LA38	
011B	206	TERMTAB LA12, TTS_LA12	
0121	207	TERMTAB LA100, TTS_LA100	
0128	208	TERMTAB LA24, TTS_LA24	
012E	209	TERMTAB LQP02, TTS_LQP02	
0135	210	TERMTAB VT55, TTS_VT55	
00	013B	.BYTE 0	
013C	212		
4E 57 4F 4E 4B 4E 55 00' 013C	213	UNKNOWN_TT: .ascic /UNKNOWN/	
07	013C		
0144	214		
0144	215	.nlist meb	
0144	216		
0144	217	.SBTTL CTSENSECHAR - Map READ_CHAR into CHAR using VMS data	
0144	218		
0144	219		
0144	220	: Inputs:	
0144	221	R0 - CTP	
0144	222	R2 - address of requested characteristics buffer (READ_CHAR)	
0144	223	R3 - address of buffer to be written	
0144	224	R9 - address of 12 buffer of current characteristics	
0144	225	R10 - address of iosb from operation	
0144	226		

		0144	227	: Outputs:					
		0144	228	: R1 is buffer length from start					
		0144	229	: buffer pointed to by R3 is complete					
		0144	230	: All registers destroyed					
		0144	231						
		0144	232	: CTSENSECHAR::					
		0144	233						
		51 50	28 2A	53 A0	DD 3C	0146	234	PUSHL R3	: Save
		51 50	51 50	50 50	C1	014E	235	MOVZWL CTP\$W_MSGSIZE(R0),R1	: message size
						014E	236	MOVAB CTP\$B_MSGTYPE(R0),R0	: base address
						0152	237	ADDL3 R0,R1,R11	: add together for end of message address
						0152	238		
						0152	239	10\$:	
		56 5B	82 52	3C D1	0152 0155	240 241	MOVZWL (R2)+,R6	: Get parameter	
		11	1A	0158	015A	242	CMPL R2,R1	: past end? *** should be one up?	
		83 57	56 FE9F	B0 CF	015A 015D	243 244	BGTRU 25\$	: Done, exit	
						0162	245	MOVW R6,(R3)+	: Move into table
		87 07	56 13	B1 0165	0162 0165	246 247	MOVAB SENSE_TABLE,R7	: Get table address	
		87 F7	85 18	0167 0169	0162 0168	248 249	CMPW R6,(R7)+	: Compare to table entry	
						0168	250	BEQL 30\$	: Branch if match found
						0168	251	TSTW (R7)+	: Advance pointer past routine
						0168	252	BGEQ 20\$	: Loop if greater than or equal to zero
						016E	253	: PANIC	: 'invalid sense mode returned'
		0193	31	016B	016B	254	BRW POST_SENSE_EXIT	: exit code	
						016E	255		
						016E	256	: Dispatch to routine	
						016E	257		
						016E	258	30\$:	
56	0000017F'EF	9E	016E	016E	259	MOVAB SENSE_ROUTINES,R6	: Base		
	57 67	3C	0175	260	MOVZWL (R7),R7	: get routine address			
	57 56	C0	0178	261	ADDL2 R6,R7	: Add offset to base for routine address			
	67 D3	16 11	017B 017D	262 263	JSB (R7)	: jsb to routine			
					BRB 10\$	: Do next parameter			
						017F	264		
						017F	265		
						017F	266	: All these routines have R0,R1,R4,R6-R8 as scratch	
						017F	267		
						017F	268		

017F 270 SENSE\_ROUTINES:  
 017F 271  
 017F 272 ; physical  
 017F 273  
 50 03 AA 9A 017F 274 INPUT\_SPEED: ; physical 1  
 06 13 0183 275 MOVZBL TT\_IOSB+3(R10),R0 ; Get speed  
 0E 10 0185 276 BEQL OUTPUT SPEED ; If zero, then same as output speed  
 83 50 80 0187 277 BSBB GET SPEED ; map from TT\$C\_BAUD\_xxxx  
 05 018A 278 MOVW R0,(R3)+ ; set receive speed  
 018B 279 RSB ; Return  
 50 02 AA 9A 018B 281 OUTPUT\_SPEED: ; physical 2  
 04 10 018F 282 MOVZBL TT\_IOSB+2(R10),R0 ; set transmit speed  
 83 50 80 0191 283 BSBB GET SPEED ; map from TT\$C\_BAUD\_xxxx  
 05 0194 284 MOVW R0,(R3)+ ; set transmit speed  
 0195 285 RSB ; Return  
 0195 286 ; Local routine to map TT\$C\_BAUD rates into speed  
 0195 287  
 0195 288  
 0195 289 ; input:  
 0195 290 R0 - Baud rate in TT\$C\_BAUD\_xxx terms  
 0195 291  
 0195 292 ; Output:  
 0195 293 R0 - baud rate in BPS  
 0195 294  
 0195 295 GET\_SPEED:  
 51 50 FEF0 CF 9E 0195 296 MOVAB SENSE BAUD,R1 ; Get table  
 6140 3C 019A 297 MOVZWL (R1)(R0),R0 ; Index in and fetch value  
 05 019E 298 RSB ; Return  
 019F 299  
 50 07 90 019F 300 CHARACTER\_SIZE: ; physical 3  
 0F E1 01A2 301 MOVVB #7,R0 ; Character size  
 03 04 A9 01A4 302 BBC #TT\$V\_EIGHTBIT-  
 50 08 90 01A7 303 TT\_CHAR1(R9),10\$ ; check characteristic  
 01AA 304 MOVVB #8,R0 ; Character size  
 83 50 98 01AA 305  
 05 01AD 306 10\$: MOVZBW R0,(R3)+ ; Set value  
 01AE 307 RSB ; Return  
 01AE 308  
 50 06 AA 01AE 309 PARITY\_ENABLE: ; physical 4  
 01 94 E1 310 CLR B R0 ; Set noparity  
 01B0 311 BBC #TT\$V\_PARITY-  
 03 06 01 90 01B2 312 TT\_IOSB+6(R10),10\$ ; Check parity  
 0185 313 MOVVB #1,R0 ; Set parity enabled  
 83 50 90 0188 314 10\$: MOVVB R0,(R3)+  
 05 0188 315 RSB ; Return  
 018C 316  
 018C 317  
 50 01 90 018C 318 PARITY\_TYPE: ; physical 5  
 07 E1 01BF 319 MOVVB #1,R0 ; Assume even  
 03 06 AA 01C1 320 BBC #TT\$V\_ODD-  
 50 02 90 01C4 321 TT\_IOSB+6(R10),10\$ ; branch if even  
 01C7 322 MOVVB #2,R0 ; Set odd  
 83 50 98 01C7 323 10\$: MOVZBW R0,(R3)+  
 05 01CA 324 RSB ; Set parity  
 01CB 325  
 01CB 326

03 04 A9	50 15	94 E1	01CB 01CB	327 MODEM_PRESENT: 328 CLRB R0 329 BBC #TT\$V MODEM,-	; physical 6 ; Get Boolean
	50 01	90	01D2	330 TT_CHAR1(R9),10\$	; check modem bit
			01D5	331 MOVB #1,R0	
83 50	90	01D5	332 10\$: MOVB R0,(R3)+	; Set value	
	05	01D8	333 RSB	; Return	
			01D9	334	
			01D9	335 AUTOBAUD_DETECT: 336 CLRB R0	; physical 7
03 08 A9	50 01	94 E1	01D9 01DB	337 BBC #TT2\$V AUTOBAUD,-	; Set autobaud
50 01	90	01E0	338 TT_CHAR2(R9),10\$		
83 50	90	01E3	339 10\$: MOVB #1,R0		
	05	01E6	340 MOVB R0,(R3)+	; Return	
			01E7	341 RSB	
83	94	01E7	342 01E7	343 MANAGEMENT_GUARANTEED: 344 CLRB (R3)+	; physical 8
	05	01E9	345 10\$: RSB	; clear ; Return	
			01EA	346	
0111	30	01EA	347 01EA	348 SWITCH_CHAR 1: 349 BSBW IGNORE_STRING	; physical 9 ; Ignore this string data
	05	01ED	350 01ED	351 RSB	; Return
			01EE	352	
010D	30	01EE	353 01EE	354 SWITCH_CHAR 2: 355 BSBW IGNORE_STRING	; physical 10 ; Ignore this string data
	05	01F1	356 01F2	356 RSB	; Return

		01F2	358	: logical		
		01F2	359			
		01F2	360	MODE_WRITING_ALLOWED:		
83	94	01F2	361	CLRB (R3)+ ; logical 1		
	05	01F4	362	10\$ RSB ; Return		
		01F5	363			
		01F5	364	TERMINAL_TYPE: ; logical 2		
		01F5	365			
		01F5	366	: Bits are:		
		01F5	367	0 - known/unknown		
		01F5	368	1 - scope/hardcopy		
		01F5	369	:		
	50	D4	01F5	370	CLRL R0	
01	A9	95	01F7	371	ASSUME TTS UNKNOWN EQ 0 ; assume unknown = 0	
	03	13	01F7	372	TSTB TT BUF+1(R9) ; is it unknown?	
50	01	88	01FC	373	BEQL 10\$ ; Branch if unknown	
		01FF	374	BISB #CTPSM_CH_KNOWN, R0 ; Set known bit		
03	04	A9	0201	375	10\$: BBC #TT\$V_SCOPE -	
50	02	88	0204	377	TT CHAR1(R9), 20\$ ; check for scope	
		0207	378	BISB #CTPSM_CH_SCOPE, R0 ; set scope		
83	50	98	0207	379	20\$: MOVZBW R0, (R3)+ ; set value	
		05	020A	380	RSB ; Return	
		020B	381			
		020B	382			
		020B	383	TERMINAL_SUBTYPE: ; logical 3		
		020B	384			
51	50	01	A9	020B	385	MOVZBL TT BUF+1(R9), R0 ; Fetch terminal type byte
	FEA5	CF	9A	020F	386	MOVAB TERM_TABLE, R1 ; Address of table
		0214	387	20\$: MOVZBL (R1)+, R4 ; Get terminal type from table		
54	81	9A	0214	388	BEQL 30\$ ; if zero, end of list => unknown	
	0D	13	0217	389	CMPB R0, R4 ; Compare terminal types	
54	50	91	0219	390	BEQL 40\$ ; Match, branch	
	0D	13	021C	391	MOVZBL (R1)+, R4 ; get length of string	
54	81	9A	021E	392	ADDL R4, R1 ; Add to pointer	
51	54	CO	0221	393	BRB 20\$ ; Loop	
	EE	11	0224	394		
		0226	395	30\$: MOVAB UNKNOWN_TT, R1 ; "unknown"		
51	FF12	CF	9E	0226	396	
		022B	397	40\$: PUSHR #^M<R2, R4, R5> ; Save registers		
54	34	BB	022B	398	MOVZBL (R1), R4 ; Get length	
	61	9A	022D	399	INCL R4 ; Include byte for count	
63	61	54	D6	0230	400	MOVC3 R4, (R1), (R3) ; Move to buffer, update R3
	34	54	28	0232	401	POPR #^M<R2, R4, R5> ; restore
		BA	0236	402		
		0238	403			
		05	0238	404	RSB ; Return	
		0239	405			
		0239	406	OUTPUT_FLOW_CONTROL: ; logical 4		
		0239	407	CLRB R0 ; assume no		
03	04	A9	023B	408	BBC #TT\$V_TTSYNC, -	
	50	01	90	0240	409	TT CHAR1(R9), 10\$ ; branch if not tt sync
		0243	410	MOVB #1, R0 ; Set true		
83	50	90	0243	411	10\$: MOVB R0, (R3)+ ; Set data	
		05	0246	412	RSB ; Return	
		0247	413			
			414			

			0247	415	OUTPUT_PAGE_STOP:		
			50 94 0247	416	CLRB RO		; logical 5
			0E E1 0249	417	BBC #TT\$V HOLDSCREEN,-		; assume no
03 04	A9		50 01 0248	418	TT_CHAR1(R9),10\$		
			50 01 024E	419	MOV B #1,RO		; branch if not holdscreen
			83 50 0251	420	10\$: MOV B R0,(R3)+		; Set true
			05 0254	421	RSB		
			0255	422			; Set data
			0255	423			; Return
			83 94 0255	424	FLOW_CHAR_PASS_THRU:		
			05 0257	425	CLRB (R3)+		; logical 6
			0258	426	10\$: RSB		
			0258	427			; Return
			50 94 0258	428	INPUT_FLOW_CONTROL:		
			04 E1 025A	429	CLRB RO		; logical 7
03 04	A9		50 01 025C	430	BBC #TT\$V HOSTSYNC -		; assume no
			50 01 025F	431	TT_CHAR1(R9),10\$		
			83 50 0262	432	MOV B #1,RO		; branch if not host sync
			05 0265	433	10\$: MOV B R0,(R3)+		; Set true
			0266	434	RSB		
			83 94 0266	435			; Set data
			0268	436			; Return
			05 0268	437	LOSS_NOTIFICATION:		
			0268	438	CLRB (R3)+		; logical 8
			05 0268	439	10\$: RSB		; ignore
			0269	440			
			0269	441			; Bell on loss data?
83	02 A9		0269	442	LINE_WIDTH:		
			05 026D	443	MOV W TT_BUF+2(R9),(R3)+		; logical 9
			026E	444	RSB		; Line width
			026E	445			; Return
83	07 A9		026E	446	PAGE_LENGTH:		
			05 0272	447	MOV ZBW TT_CHAR1+3(R9),(R3)+		; logical 10
			0273	448	RSB		; Page length
			0273	449			; Return
83	B4		0273	450	STOP_LENGTH:		
			05 0275	451	CLR W (R3)+		; logical 11
			0276	452	RSB		; ** ignore
			0276	453			; Return
			50 B4 0276	454	CR_FILL:		
			0A E1 0278	455	CLR W RO		; logical 12
50	04 04	A9	027A	456	BBC #TT\$V CRFILL,-		; Assume no cr fill
			06 AA 027D	457	TT_CHAR1(R9),10\$		
			0281	458	MOV W TT_IOSB+6(R10),R0		; branch if off
			83 50 0281	459	10\$: MOV W TT_IOSB+6(R10),R0		; set on
			05 0284	460	MOV W RO,(R3)+		
			0285	461	RSB		
			0285	462			; Return
			50 B4 0285	463	LF_FILL:		
			0B E1 0287	464	CLR W RO		; logical 13
50	04 04	A9	0289	465	BBC #TT\$V LFFILL,-		; Assume no lf fill
			07 AA 028C	466	TT_CHAR1(R9),10\$		
			0290	467	MOV W TT_IOSB+7(R10),R0		; branch if off
			83 50 0290	468	10\$: MOV W TT_IOSB+7(R10),R0		; set on
			05 0293	469	MOV W RO,(R3)+		
			0294	470	RSB		
			0294	471			; Return

			0294	472	WRAP:		
50	01	B0	0294	473	MOVW	#1, R0	; logical 14
09		E1	0297	474	BBC	#TTSV WRAP -	; assume no wrap
03 04	A9		0299	475		TT CHAR1(R9),10\$	; branch if no wrap
50	04	B0	029C	476	MOVW	#4, R0	; Set wrap
			029F	477	10\$:		
83	50	B0	029F	478	MOVW	R0,(R3)+	; Set data
	05		02A2	479	RSB		; Return
			02A3	480			
			02A3	481	HORIZONTAL TAB:		
50	02	B0	02A3	482	MOVW	#2, R0	; logical 15
08		E1	02A6	483	BBC	#TTSV MECHTAB -	; Assume no mechtab
03 04	A9		02A8	484		TT CHAR1(R9),10\$	; Set mechtab
50	01	B0	02AB	485	MOVW	#1, R0	; set mechtab
			02AE	486	10\$:		
83	50	B0	02AE	487	MOVW	R0,(R3)+	; set data
	05		02B1	488	RSB		; Return
			02B2	489			
			02B2	490	VERTICAL TAB:		
83	B4	02B2	491		CLRW	(R3)+	; logical 16
	05	02B4	492		RSB		; ignore
		02B5	493				; Return
		02B5	494	FORM_FEED:			
50	02	B0	02B5	495	MOVW	#2, R0	; logical 17
13		E1	02B8	496	BBC	#TTSV MECHFORM -	; Assume no mechform
03 04	A9		02BA	497		TT CHAR1(R9),10\$	; Set mechform
50	01	B0	02BD	498	MOVW	#1, R0	; set mechform
			02C0	499	10\$:		
83	50	B0	02C0	500	MOVW	R0,(R3)+	; set data
	05		02C3	501	RSB		; Return
			02C4	502			
			02C4	503			

```

02C4 505 ; cterm
02C4 506
02C4 507 IGNORE_INPUT: ; cterm 1
05 02C4 508 CLR B (R3)+ ; Ignore
02C6 509 10$: RSB ; Return
02C7 510
02C7 511 CHAR_ATTRIBUTES: ; cterm 2
02C7 512
02C7 513
05 02C7 514 RSB
02C8 515
83 94 02C8 516 CONTROL_O_PASS_THRU: ; cterm 3
05 02C8 517 CLR B (R3)+ ; Ignore
02CA 518 10$: RSB ; Return
02CB 519
02CB 520 RAISE_INPUT: ; cterm 4
50 94 02CB 521 CLR B R0 ; Assume lower case (i.e. NO RAISE)
07 E0 02CD 522 BBS #TT$V LOWER,-
03 04 A9 02CF 523 TT_CHAR1(R9),10$ ; Branch if NO RAISE
50 01 90 02D2 524 MOVB #1,R0 ; Set RAISE
02D5 525 10$: ; set data
83 50 90 02D5 526 MOVB R0,(R3)+ ; Return
05 02D8 527 RSB
02D9 528
50 94 02D9 529 NORMAL_ECHO: ; cterm 5
01 E0 02DB 530 CLR B R0 ; (note opposite sense)
03 04 A9 02DD 531 BBS #TT$V NOECHO,-
50 01 90 02E0 532 TT_CHAR1(R9),10$ ; check noecho
02E3 533 MOVB #1,R0
83 50 90 02E3 534 10$: ; set data
05 02E6 535 MOVB R0,(R3)+ ; Return
02E7 536 RSB
02E7 537
02E7 538 :*** REQUEST_PROCESSING: ; cterm 6
02E7 539
02E7 540 INPUT_ESCAPE_ENABLE: ; cterm 7
02E7 541 OUTPUT_ESCAPE_ENABLE: ; cterm 8
50 94 02E7 542 CLR B R0
03 E1 02E9 543 BBC #TT$V ESCAPE,-
03 04 A9 02EB 544 TT_CHAR1(R9),10$ ; Set escape
50 01 90 02EE 545 MOVB #1,R0
02F1 546 10$: ; Return
83 50 90 02F1 547 MOVB R0,(R3)+ ; Return
05 02F4 548 RSB
02F5 549
83 B4 02F5 550 INPUT_COUNT_STATE: ; cterm 9
05 02F7 551 CLR B (R3)+ ; Ignore
02F8 552 RSB
02F8 553
83 94 02F8 554 AUTO_PROMPT: ; cterm 10
05 02FA 555 CLR B (R3)+ ; Ignore
02FB 556 10$: RSB ; Return
02FB 557
83 94 02FB 558 ERROR_PROCESSING: ; cterm 11
05 02FD 559 CLR B (R3)+ ; Ignore
02FE 560 10$: RSB ; Return
02FE 561

```

- RTPAD/CTERM SENSE CHARACTERISTICS<sup>D 12</sup>  
CTSENSECHAR - Map READ\_CHAR into CHAR us 16-SEP-1984 02:11:02 VAX/VMS Macro V04-00  
5-SEP-1984 03:14:55 [RTPAD.SRC]CTSENSE, MAR;1 Page 12  
(5)

83 94 02FE 562  
05 02FE 563 IGNORE\_STRING:  
0300 564 CLRB (R3)+  
0301 565 RSB  
0301 566

0301 568 POST\_SENSE\_EXIT:  
0301 569  
0301 570  
0301 571 : Return R1 as correct length of characteristics  
0301 572  
51 53 51 8ED0 0301 573 POPL R1  
C3 0304 574 SUBL3 R1,R3,R1 ; start of buffer  
05 0308 575 RSB ; R1 = R3 - R1  
0309 576  
0309 577 .end

AUTOBAUD\_DETECT  
 AUTO\_PROMPT  
 CHSC\_CTERM  
 CHSC\_LOGICAL  
 CHSC\_PHYSICAL  
 CHARACTER\_SIZE  
 CHAR\_ATTRIBUTES  
 CHAR\_COUNT  
 CHAR\_TYPE  
 CONTROL\_O\_PASS\_THRU  
 CR\_FILL  
 CTERM  
 CTPSB\_MSGTYPE  
 CTPSM\_CH\_KNOWN  
 CTPSM\_CH\_SCOPE  
 CTPSW\_MSGSIZE  
 CTSENSECHAR  
 ERROR\_PROCESSING  
 FLOW\_CHAR\_PASS\_THRU  
 FORM\_FEED  
 GET\_SPEED  
 HORIZONTAL\_TAB  
 IGNORE\_INPUT  
 IGNORE\_STRING  
 INPUT\_COUNT\_STATE  
 INPUT\_ESCAPE\_ENABLE  
 INPUT\_FLOW\_CONTROL  
 INPUT\_SPEED  
 LF\_FILL  
 LINE\_WIDTH  
 LOGICAL  
 LOSS\_NOTIFICATION  
 MANAGEMENT\_GUARANTEED  
 MODEM\_PRESENT  
 MODE\_Writing\_ALLOWED  
 NORMAL\_ECHO  
 OUTPUT\_ESCAPE\_ENABLE  
 OUTPUT\_FLOW\_CONTROL  
 OUTPUT\_PAGE\_STOP  
 OUTPUT\_SPEED  
 PAGE\_LENGTH  
 PARITY\_ENABLE  
 PARITY\_TYPE  
 PHYSICAL  
 POST\_SENSE\_EXIT  
 RAISE\_INPUT  
 SENSE\_BAUD  
 SENSE\_ROUTINES  
 SENSE\_TABLE  
 STOP\_LENGTH  
 SWITCH\_CHAR\_1  
 SWITCH\_CHAR\_2  
 TERMINAL\_SUBTYPE  
 TERMINAL\_TYPE  
 TERM\_TABLE  
 TT\$V\_CRFILL  
 TT\$V\_EIGHTBIT

000001D9 R 02	TT\$V_ESCAPE	= 00000003
000002F8 R 02	TT\$V_HOLDSCREEN	= 0000000E
= 00000002	TT\$V_HOSTSYNC	= 00000004
= 00000001	TT\$V_LFFILL	= 0000000B
= 00000000	TT\$V_LOWER	= 00000007
0000019F R 02	TT\$V_MECHFORM	= 00000013
000002C7 R 02	TT\$V_MECHTAB	= 00000008
= 0000000B	TT\$V_MODEM	= 00000015
= 00000200	TT\$V_NOECHO	= 00000001
000002C8 R 02	TT\$V_ODD	= 00000007
00000276 R 02	TT\$V_PARITY	= 00000006
= 00000200	TT\$V_SCOPE	= 0000000C
= 0000002A	TT\$V_TTSYNC	= 00000005
= 00000001	TT\$V_WRAP	= 00000009
= 00000002	TT\$[A100	= 00000025
= 00000028	TT\$-LA12	= 00000024
00000144 RG 02	TT\$-LA24	= 00000025
000002FB R 02	TT\$-LA34	= 00000022
00000255 R 02	TT\$-LA36	= 00000020
000002B5 R 02	TT\$-LA38	= 00000023
00000195 R 02	TT\$-LQPO2	= 00000026
000002A3 R 02	TT\$-UNKNOWN	= 00000000
000002C4 R 02	TT\$-VK100	= 00000002
000002FE R 02	TT\$-VT05	= 00000001
000002F5 R 02	TT\$-VT100	= 00000060
000002E7 R 02	TT\$-VT101	= 00000061
00000258 R 02	TT\$-VT102	= 00000062
0000017F R 02	TT\$-VT105	= 00000063
00000285 R 02	TT\$-VT125	= 00000064
00000269 R 02	TT\$-VT131	= 00000065
= 00000100	TT\$-VT132	= 00000066
00000266 R 02	TT\$-VT173	= 00000003
000001E7 R 02	TT\$-VT52	= 00000040
000001CB R 02	TT\$-VT55	= 00000041
000001F2 R 02	TT\$V_AUTOBAUD	= 00000001
000002D9 R 02	TT-BUF	= 00000000
000002E7 R 02	TT-CHAR1	= 00000004
00000239 R 02	TT-CHAR2	= 00000008
00000247 R 02	TT-IOSB	= 00000000
0000018B R 02	UNRKNOWN TT	0000013C R 02
0000026E R 02	VERTICAL_TAB	000002B2 R 02
000001AE R 02	WRAP	00000294 R 02
= 00000000		
00000301 R 02		
000002C8 R 02		
00000096 R 02		
0000017F R 02		
00000000 R 02		
00000273 R 02		
000001EA R 02		
000001EE R 02		
0000020B R 02		
000001F5 R 02		
000000B8 R 02		
= 0000000A		
= 0000000F		

```
+-----+
! Psect synopsis !
+-----+
```

## PSECT name

	Allocation	PSECT No.	Attributes														
ABS .	00000000 ( 0.)	00 ( 0.)	NOPIC	USR	CON	ABS	LCL	NOSHR	NOEXE	NORD	NOWRT	NOVEC	BYTE				
\$ABSS	00000000 ( 0.)	01 ( 1.)	NOPIC	USR	CON	ABS	LCL	NOSHR	EXE	RD	WRT	NOVEC	BYTE				
RTPAD	00000309 ( 777.)	02 ( 2.)	NOPIC	USR	CON	REL	LCL	NOSHR	EXE	RD	NOWRT	NOVEC	BYTE				

```
+-----+
! Performance indicators !
+-----+
```

## Phase

Phase	Page faults	CPU Time	Elapsed Time
Initialization	32	00:00:00.05	00:00:01.46
Command processing	122	00:00:00.50	00:00:03.53
Pass 1	367	00:00:08.00	00:00:35.16
Symbol table sort	0	00:00:01.37	00:00:03.69
Pass 2	106	00:00:01.64	00:00:06.41
Symbol table output	13	00:00:00.07	00:00:00.07
Psect synopsis output	1	00:00:00.02	00:00:00.02
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	643	00:00:11.65	00:00:50.34

The working set limit was 1500 pages.

68631 bytes (135 pages) of virtual memory were used to buffer the intermediate code.

There were 70 pages of symbol table space allocated to hold 1274 non-local and 33 local symbols.

577 source lines were read in Pass 1, producing 16 object records in Pass 2.

17 pages of virtual memory were used to define 16 macros.

```
+-----+
! Macro library statistics !
+-----+
```

## Macro library name

## Macros defined

Macro library name	Macros defined
\$255\$DUA28:[RTPAD.OBJ]RTPAD.MLB;1	1
\$255\$DUA28:[SYS.OBJ]LIB.MLB;1	0
\$255\$DUA28:[SYSLIB]STARLET.MLB;2	8
TOTALS (all libraries)	9

1303 GETS were required to define 9 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS\$:[CTSENSERT/0BJ=0BJ\$:[CTSENSERT MSRC\$:[CTSENSERT/UPDATE=(ENH\$:[CTSENSERT)+EXECMLS/LIB+LIB\$:[RTPAD/LIB

0333 AH-BT13A-SE  
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION  
CONFIDENTIAL AND PROPRIETARY

CTDSENSE  
LIS

CTDUMSPEC  
LIS

CTSENSERT  
LIS

RSTSRT  
LIS

CTERMRT  
LIS

DTE DF03  
LIS

CTSETRT  
LIS

CTDSET  
LIS